Final Candy

2022-12-06

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(readr)
library(tseries)
IPG3113N <- read_csv("/Users/danieltamayo/Desktop/CandyFinal.csv")
```

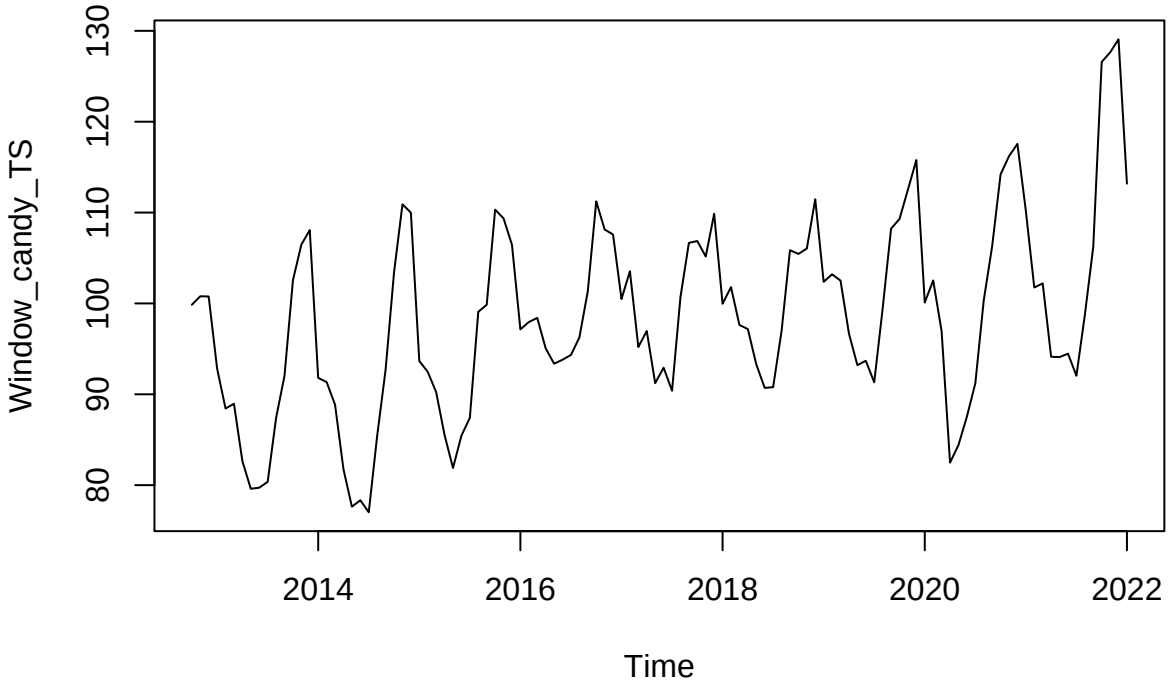
```
## Rows: 121 Columns: 2
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): IPG3113N
## date (1): DATE
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
candy_ts <- ts(IPG3113N$IPG3113N,frequency = 12,start=c(2012,10))
Window_candy_TS <-window(candy_ts, start = 2012, end = 2022)
```

```
## Warning in window.default(x, ...): 'start' value not changed
```

```
## Plot and Inference
plot(Window_candy_TS)
```



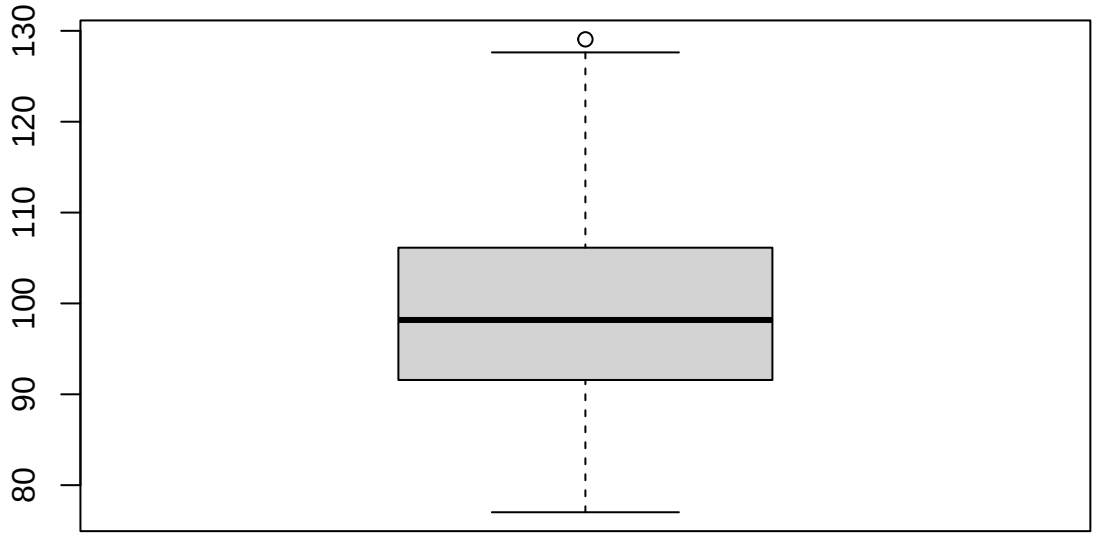
From this plot, we can see that over time from 2012 to 2022 there appears to be a seasonal component of candy production in the US. There appears to be an element of seasonality as well. This element of seasonality appears to happen once a year.

```
#Central Tendency
summary(Window_candy_TS)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   77.02   91.69   98.18   98.50  106.09  129.06
```

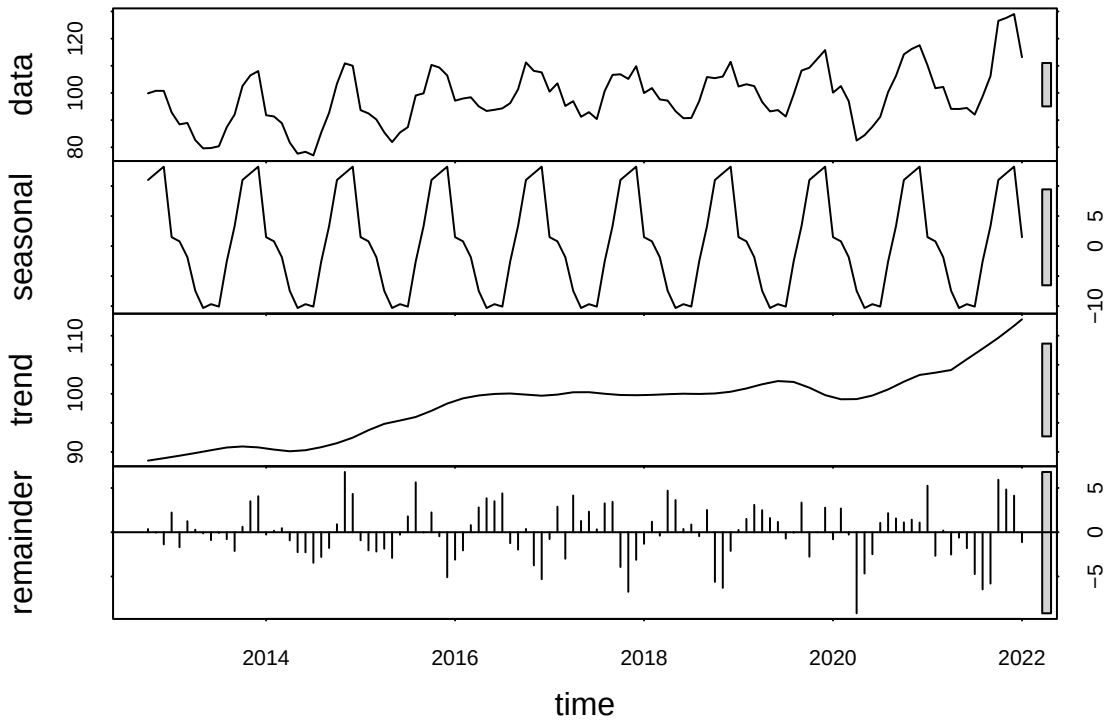
Including Plots

```
boxplot(Window_candy_TS)
```



FRom this we can see that the Data seems to be pretty unifromly distributed; The mean and the median are about the same. However most of the data lies between the min and the median/mean.

```
#Decomposition
stl_decomp <- stl(Window_candy_TS,s.window ="periodic")
plot(stl_decomp)
```



From the decomposition, we can see that there is quite a strong element of seasonality in the Candy Time Series.

Since the variation of seasonality(widths and heights) does not change over time, we can see that this decomposition is in fact additive.

```
#Decomposition
Window_candy_TS
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2012
## 2013  92.8393  88.4378  88.9572  82.6418  79.6009  79.7216  80.3623  87.3990
## 2014  91.8045  91.3464  88.8527  81.7442  77.6292  78.3359  77.0192  85.4452
## 2015  93.6676  92.4733  90.2388  85.5111  81.8907  85.4442  87.4143  99.0757
## 2016  97.1509  97.9475  98.4111  95.0657  93.3682  93.8015  94.3271  96.2487
## 2017 100.4828 103.5407  95.2095  96.9654  91.2276  92.9268  90.3917 100.6988
## 2018  99.9651 101.7932  97.6257  97.1776  93.2973  90.7030  90.7825  96.9555
## 2019 102.3790 103.2079 102.5133  96.6862  93.2061  93.6885  91.3219  99.4256
## 2020 100.1032 102.5297  96.9412  82.4777  84.4155  87.5411  91.2016 100.3417
## 2021 110.1841 101.7558 102.2030  94.1243  94.0966  94.4684  92.0391  98.7301
## 2022 113.1828
##           Sep      Oct      Nov      Dec
## 2012           99.8541 100.7874 100.7643
```

```
## 2013  92.0523 102.5585 106.4783 108.0668
## 2014  92.7103 103.4097 110.9082 109.9962
## 2015  99.8525 110.3226 109.3626 106.4516
## 2016 101.3271 111.2323 108.1348 107.5859
## 2017 106.6639 106.8689 105.1628 109.8611
## 2018 105.8597 105.4468 106.0408 111.4674
## 2019 108.2320 109.2961 112.5268 115.7811
## 2020 106.3095 114.2083 116.2062 117.5581
## 2021 106.2289 126.5753 127.6255 129.0640
## 2022
```

Here are the values of the seasonal monthly indices.

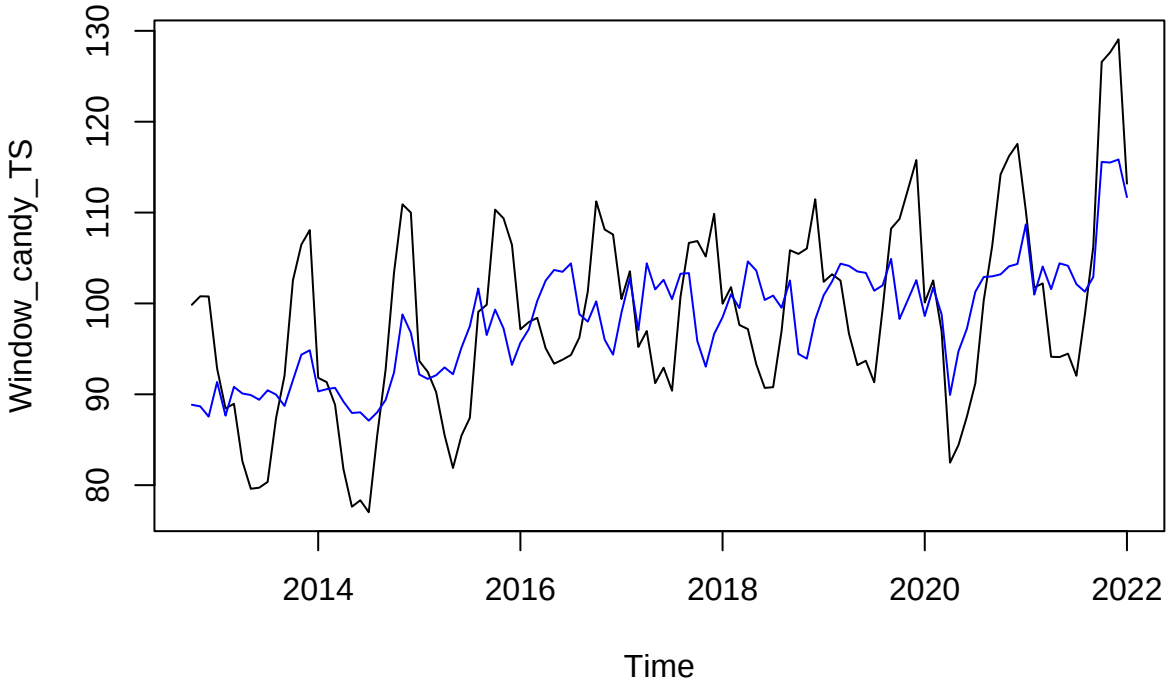
```
#Decomposition

tapply(Window_candy_TS,cycle(Window_candy_TS),mean)
```

```
##         1         2         3         4         5         6         7         8
## 100.17593  98.11470  95.66139  90.26600  87.63690  88.51456  88.31774  96.03559
##         9        10        11        12
## 102.13736 108.97726 110.32334 111.65965
```

These are the average values for the months from our Time Series. From this, we can see that based on the average, The most Candy is produced in December and the least amount of Candy is produced in May.

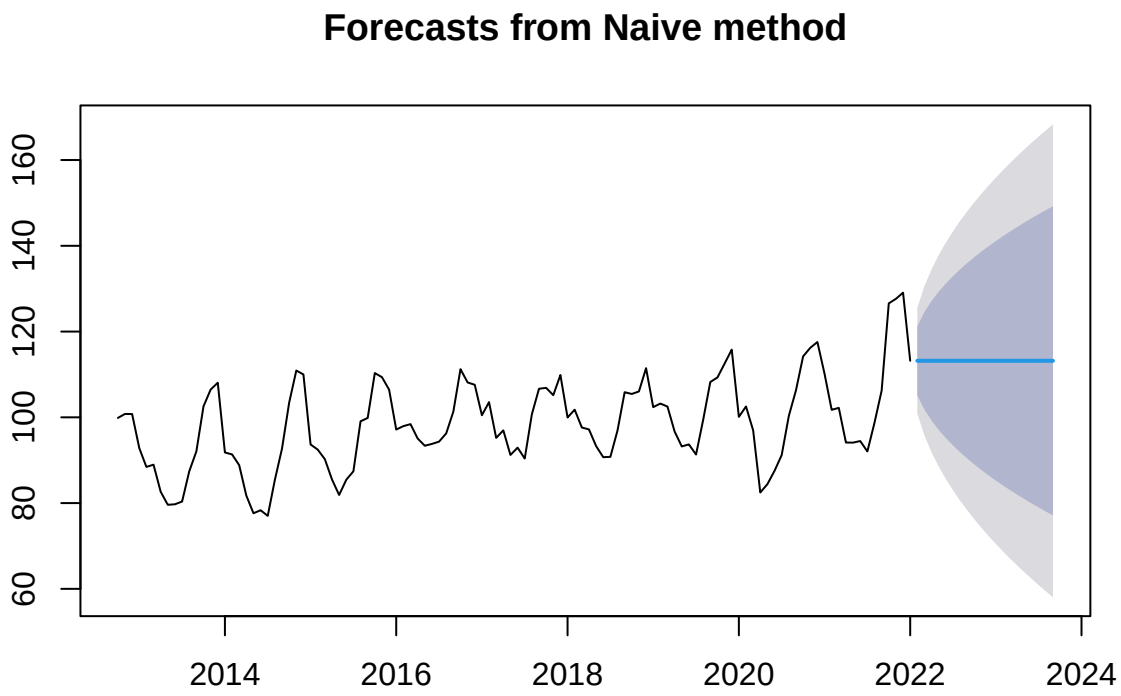
##I believe the most Candy is produced in December for Holiday season and in May there is not that much because there are not that many known festivities in May that involve candy.

```
#Decomposition
plot(Window_candy_TS)
lines(seasadj(stl_decomp), col="blue")
```

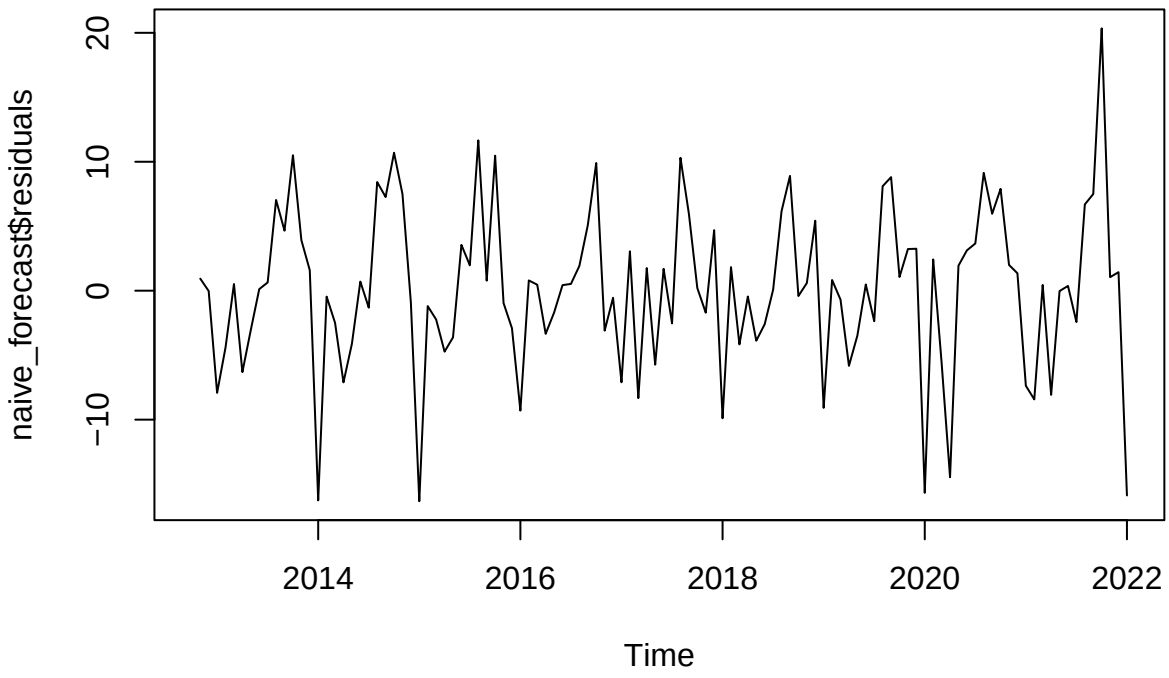


This is the graph of our regular TS and our seasonaly adjusted Time Series in Blue over that.

```
#Naive Method
naive_forecast <- naive(Window_candy_TS,20)
plot(naive_forecast)
```

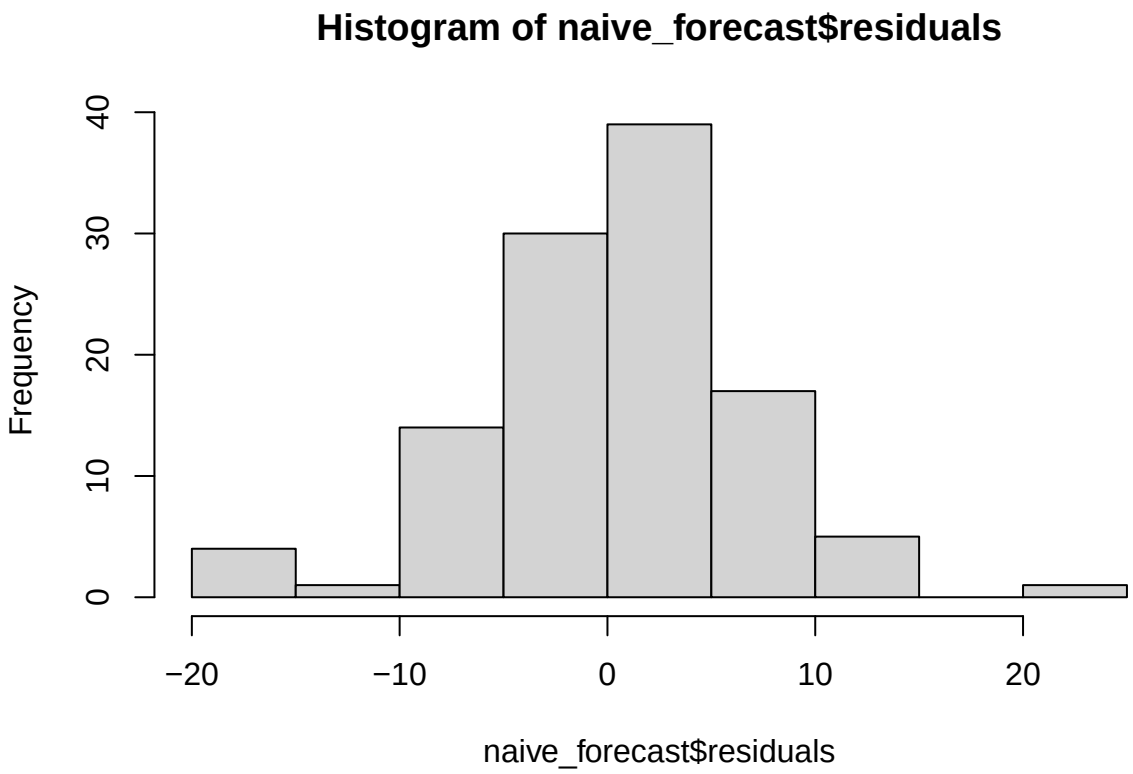



```
#Naive Method

plot(naive_forecast$residuals)
```



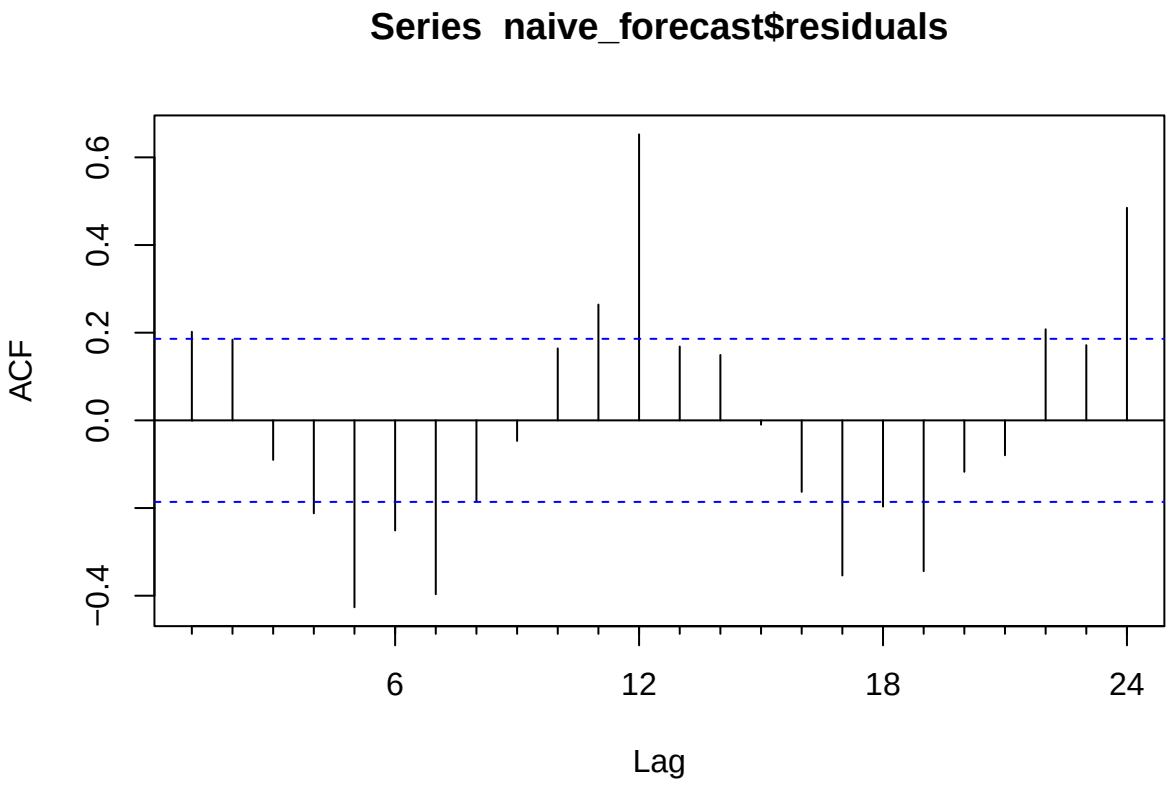
##Since there appears to be seasonlaity and a trend in the resiudals. This shows that the naive forecast is not a good fit for this data set.

```
#Naive Method

hist(naive_forecast$residuals)
```



##This shows that he data is pretty normally distributed.

```
#Naive Method

Acf(naive_forecast$residuals)
```



##This indicates that there seems to be a seasonal component, there are also some values in the residuals which are statistically significant.

```
#Naive Method

accuracy(naive_forecast)
```

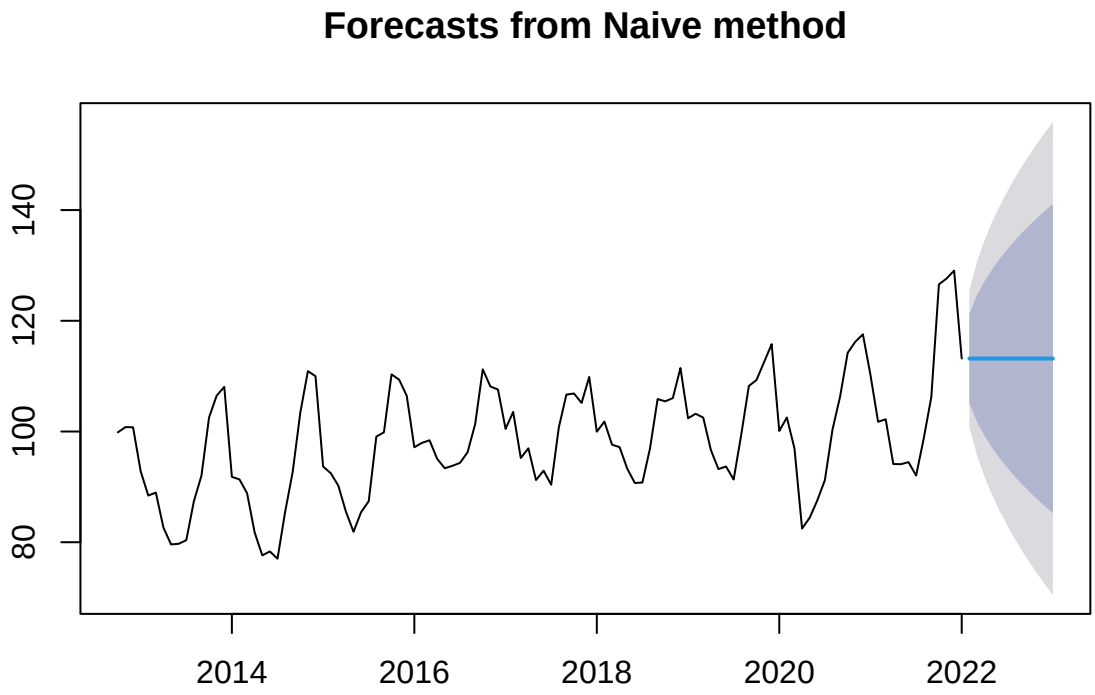
```
##                      ME     RMSE      MAE         MPE     MAPE     MASE
## Training set 0.1200784 6.290507 4.634619 -0.08377488 4.696548 1.195201
##                    ACF1
## Training set 0.2019946
```

```
#Naive Method


naive_forecast_12 <- naive(Window_candy_TS,12)
naive_forecast_12
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Feb 2022       113.1828 105.12119 121.2444 100.85363 125.5120
## Mar 2022       113.1828 101.78196 124.5836  95.74672 130.6189
## Apr 2022       113.1828  99.21968 127.1459  91.82805 134.5375
## May 2022       113.1828  97.05958 129.3060  88.52446 137.8411
## Jun 2022       113.1828  95.15649 131.2091  85.61394 140.7517
## Jul 2022       113.1828  93.43597 132.9296  82.98263 143.3830
## Aug 2022       113.1828  91.85379 134.5118  80.56289 145.8027
```

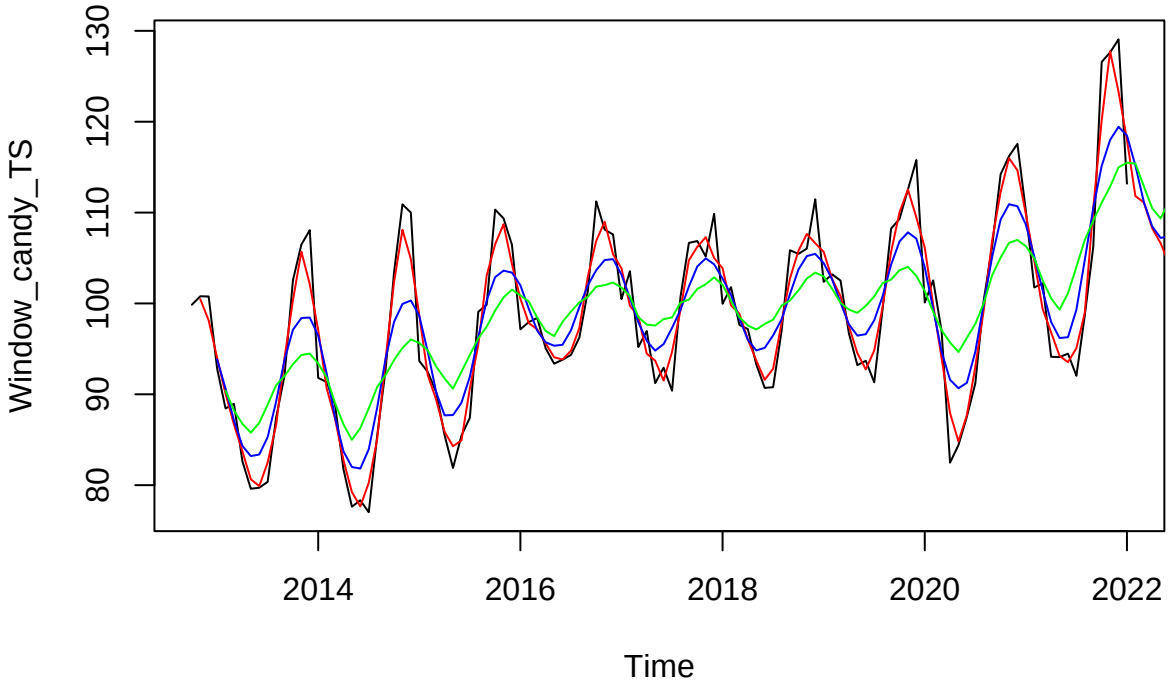
```
## Sep 2022       113.1828  90.38112 135.9845  78.31065 148.0550
## Oct 2022       113.1828  88.99797 137.3676  76.19530 150.1703
## Nov 2022       113.1828  87.68975 138.6758  74.19455 152.1711
## Dec 2022       113.1828  86.44547 139.9201  72.29158 154.0740
## Jan 2023       113.1828  85.25657 141.1090  70.47331 155.8923
```

```
#Naive Method



plot(naive_forecast_12)
```



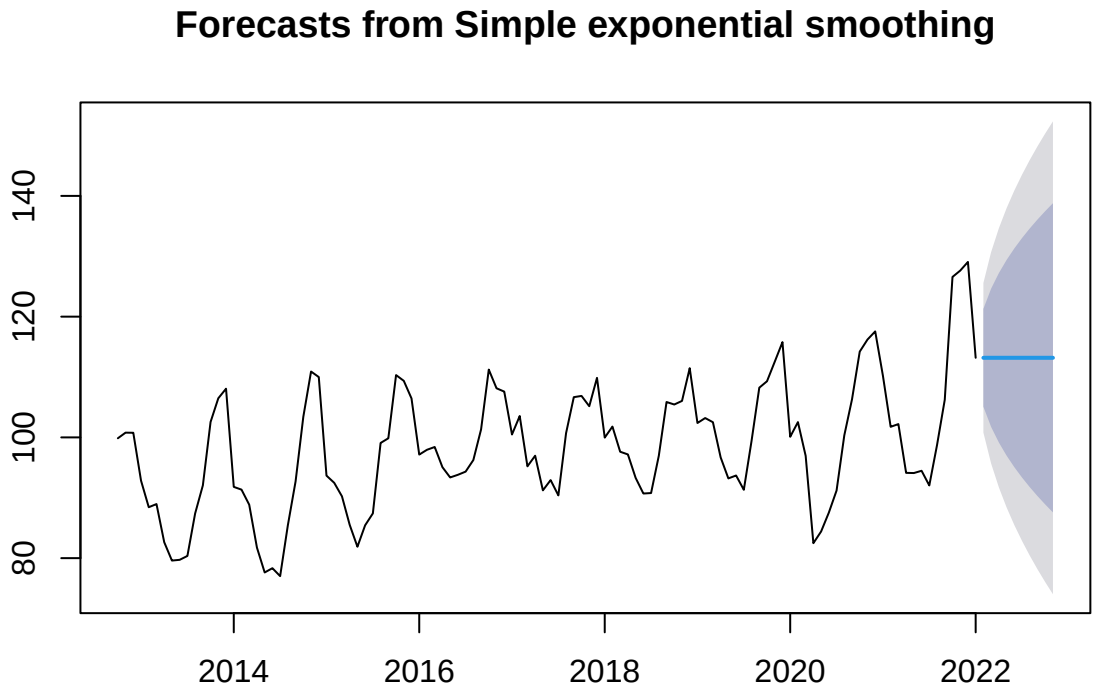
##Overall, the Naive forecast is not the best since it does not take into account seasonality and teh trend. It says the value will be 113.

```
#Simple Moving Averages

MA3_forecast <- ma(candy_ts,order=3)
MA6_forecast <- ma(candy_ts,order=6)
MA9_forecast <- ma(candy_ts,order=9)

plot(Window_candy_TS)
lines(MA3_forecast,col="Red")
lines(MA6_forecast,col="Blue")
lines(MA9_forecast,col="Green")
```



As the moving average goes up we can see that the seasonality is less wide. Overall, this smooths the graph out.

```
#Simple Smoothing
SES_candy <- ses(Window_candy_TS)
SES_candy_forecast <- forecast(SES_candy)
plot(SES_candy_forecast)
```




```
#Simple Smoothing

SES_candy
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Feb 2022       113.1844 105.08605 121.2827 100.79905 125.5697
## Mar 2022       113.1844 101.73219 124.6366  95.66976 130.6990
## Apr 2022       113.1844  99.15860 127.2102  91.73379 134.6350
## May 2022       113.1844  96.98893 129.3798  88.41558 137.9532
## Jun 2022       113.1844  95.07741 131.2914  85.49216 140.8766
## Jul 2022       113.1844  93.34925 133.0195  82.84917 143.5196
## Aug 2022       113.1844  91.76005 134.6087  80.41868 145.9501
## Sep 2022       113.1844  90.28084 136.0879  78.15644 148.2123
## Oct 2022       113.1844  88.89155 137.4772  76.03169 150.3371
## Nov 2022       113.1844  87.57751 138.7913  74.02205 152.3467
```

```
#Simple Smoothing

SES_candy$alpha
```

```
## NULL
```

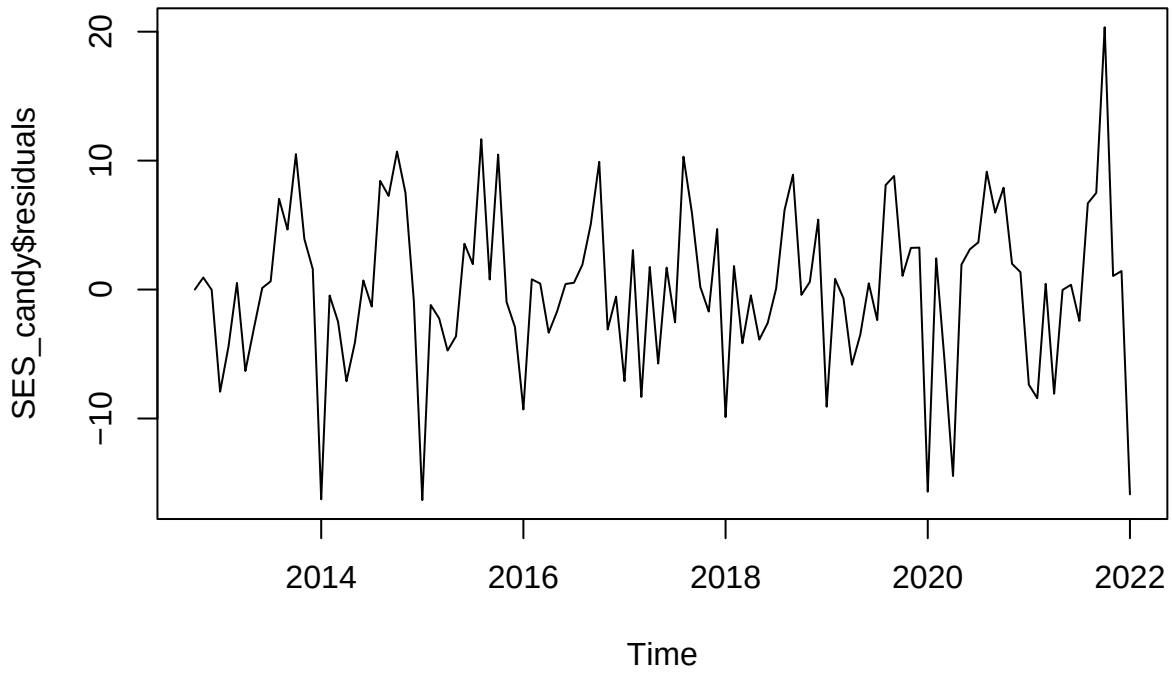
```
#Simple Smoothing

SES_candy$beta
```

```
## NULL
```

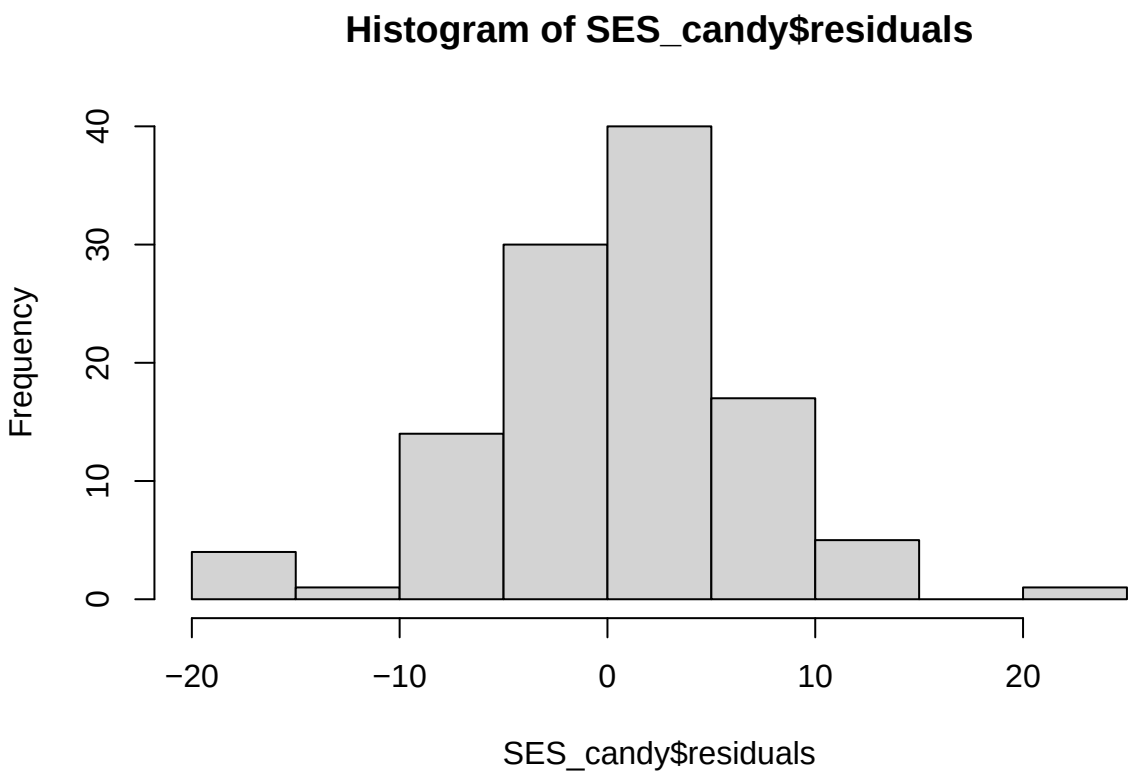
```
#Simple Smoothing

SES_candy$gamma
```

```
## NULL
```

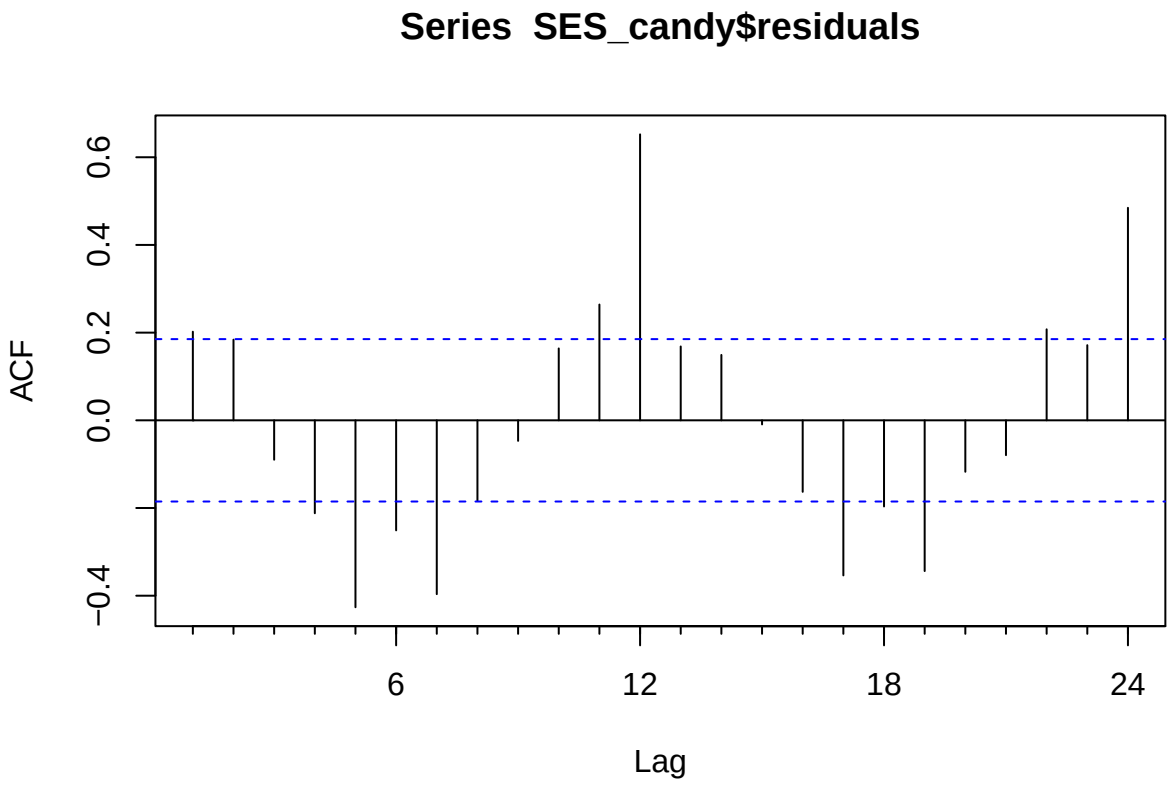
```
#Simple Smoothing

plot(SES_candy$residuals)
```




```
#Simple Smoothing

hist(SES_candy$residuals)
```

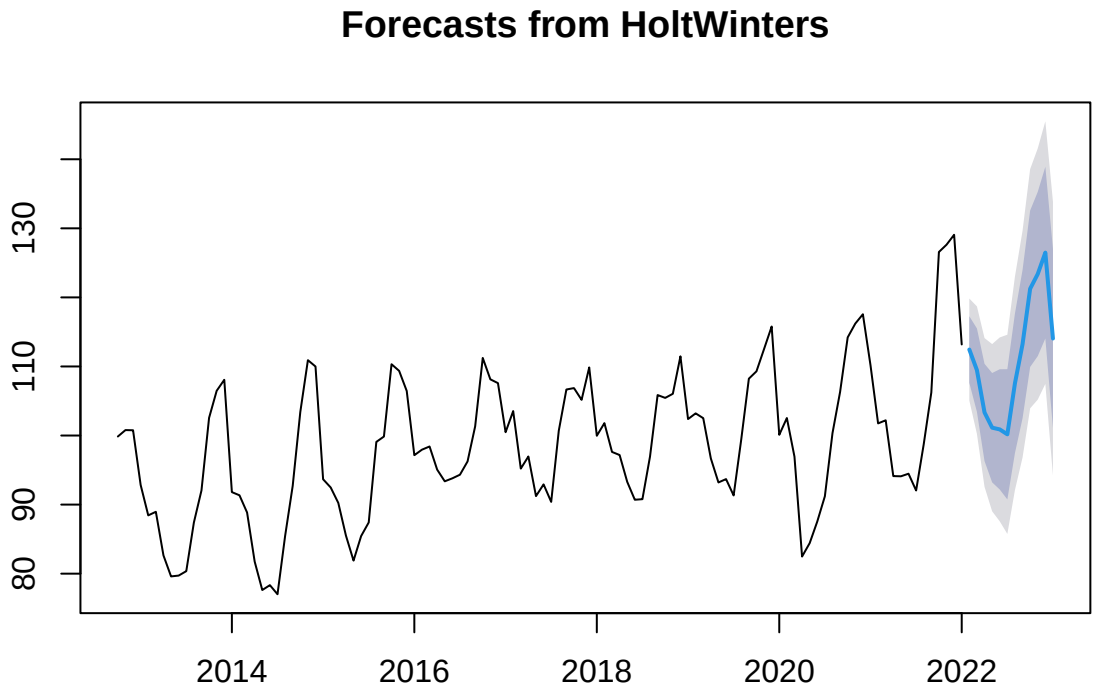



```
#Simple Smoothing

Acf(SES_candy$residuals)
```



##Overall not good- seasonality and significant values

```
#Holt Winters
HW_candy <- HoltWinters(Window_candy_TS)
HW_candy_forecast <- forecast(HW_candy,h=12)
plot(HW_candy_forecast)
```




```
#Holt Winters
HW_candy
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = Window_candy_TS)
##
## Smoothing parameters:
##  alpha: 0.7506546
##  beta : 0
##  gamma: 0.4883202
##
## Coefficients:
##             [,1]
## a   110.70009616
## b     0.02735169
## s1    1.72180742
## s2   -1.28486516
## s3   -7.43820880
## s4   -9.67567517
## s5   -9.94871899
## s6  -10.69053896
## s7   -3.32792838
## s8    2.30623793
## s9   10.30389812
```

```
## s10  12.41027415
## s11  15.47849169
## s12   3.01602423
```

From this we can see that value of alpha is 0.75, the value of beta is 0 and the value of gamma is 0.488.

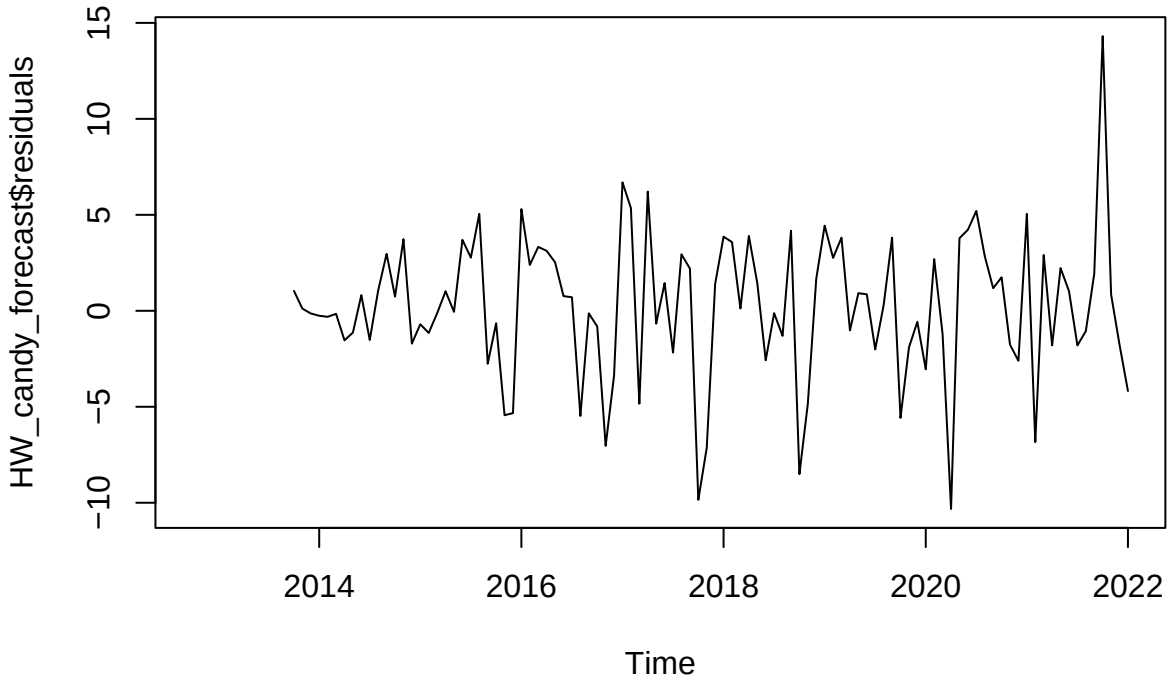
##Alpha signifies the level of the smoothing - since it is .75, later values have more weight. ##Beta signifies the level for trend smoothing - since it is 0, earlier values have all teh weight. ##Gamma refers to the seasonal component of the timeseries sinec it is .48 this means that it is right in the middle of earlier and later values.

```
#Holt Winters
HW_candy$Sigma
```

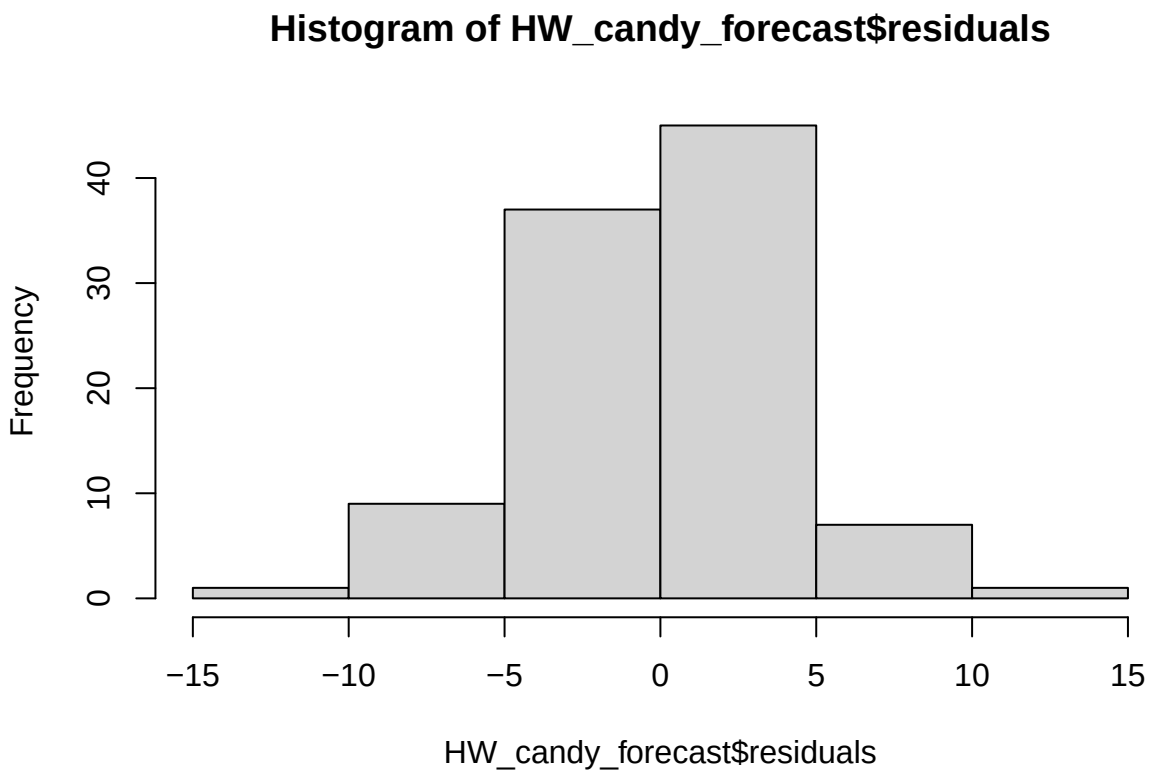
```
## NULL
```

##The value of sigma is null

```
#ARIMA
plot(HW_candy_forecast$residuals)
```

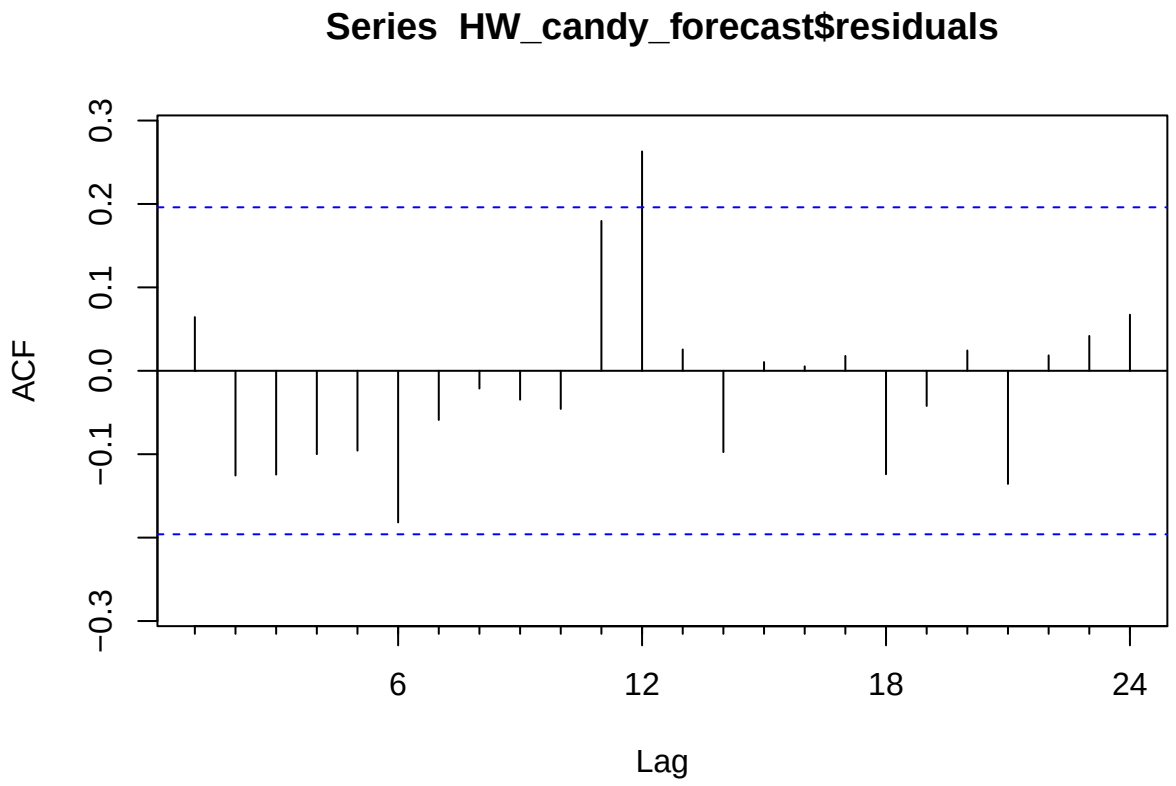


```
#ARIMA
hist(HW_candy_forecast$residuals)
```



This shows there is a normal distribution in residuals

```
#HW
Acf(HW_candy_forecast$residuals)
```

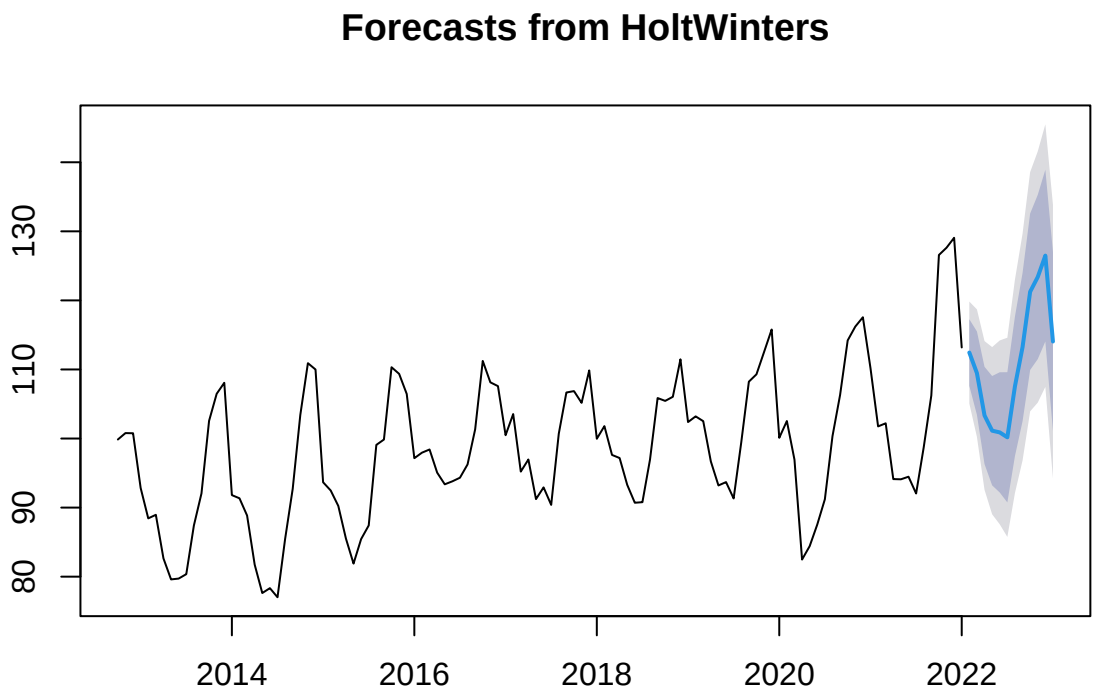


THis looks good, no trends, seasonality, all points are not significant

```
#HW

accuracy(HW_candy_forecast)
```

```
##                       ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
## Training set 0.2357743 3.753584 2.82354 0.234517 2.80643 0.7281501 0.06434792
```

```
#HW

plot(forecast(HW_candy_forecast, h=12))
```

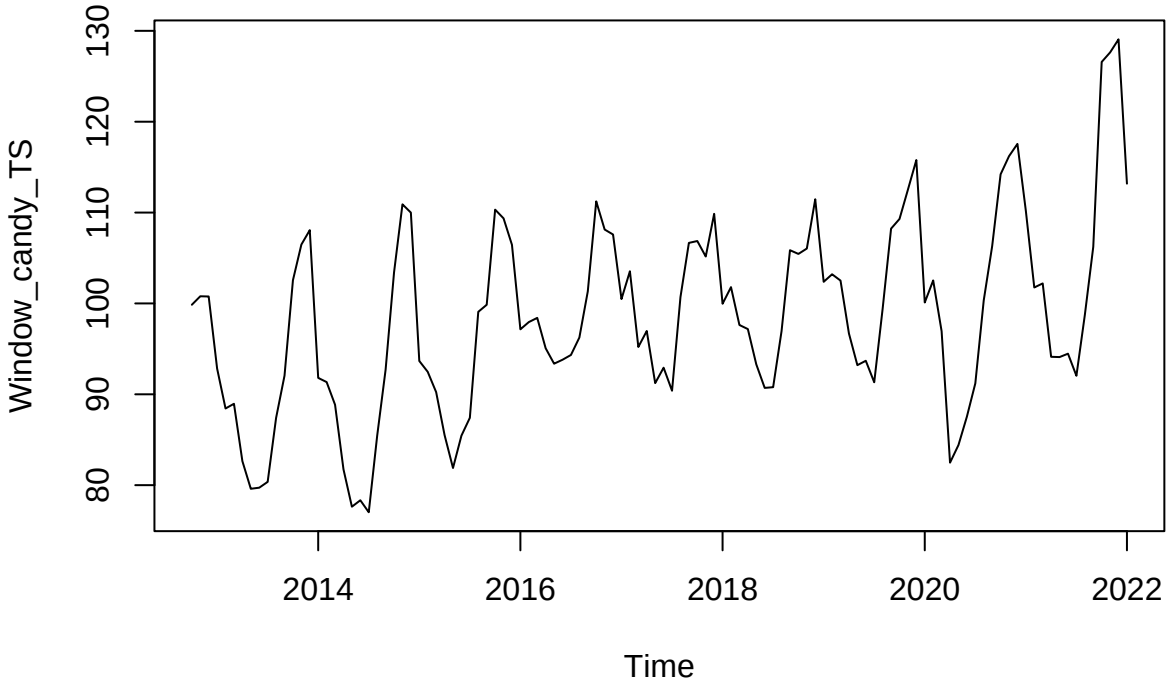



```
#HW

forecast(HW_candy_forecast, h=12)
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Feb 2022       112.4493 107.62416 117.2744 105.06990 119.8286
## Mar 2022       109.4699 103.43667 115.5032 100.24285 118.6970
## Apr 2022       103.3439  96.30696 110.3809  92.58181 114.1061
## May 2022       101.1338  93.21942 109.0482  89.02979 113.2379
## Jun 2022       100.8881  92.18431 109.5920  87.57678 114.1995
## Jul 2022       100.1737  90.74630 109.6010  85.75575 114.5916
## Aug 2022       107.5636  97.46442 117.6628  92.11822 123.0090
## Sep 2022       113.2251 102.49608 123.9542  96.81646 129.6338
## Oct 2022       121.2502 109.92622 132.5741 103.93169 138.5686
## Nov 2022       123.3839 111.49480 135.2730 105.20110 141.5667
## Dec 2022       126.4795 114.05090 138.9080 107.47161 145.4873
## Jan 2023       114.0443 101.09877 126.9899  94.24580 133.8429
```

##Overall, we can see that this a very good model. The model in a year is predicted to be 114.0443. We can tell this is a good model by the residuals.

```
#ARIMA
plot(Window_candy_TS)
```




```
#ARIMA
adf.test(Window_candy_TS)
```

```
## Warning in adf.test(Window_candy_TS): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Window_candy_TS
## Dickey-Fuller = -8.3398, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(Window_candy_TS)
```

```
## Warning in kpss.test(Window_candy_TS): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  Window_candy_TS
## KPSS Level = 0.77215, Truncation lag parameter = 4, p-value = 0.01
```

The TS is stationary since when we run the KPSS test, the p-value is less than .05.

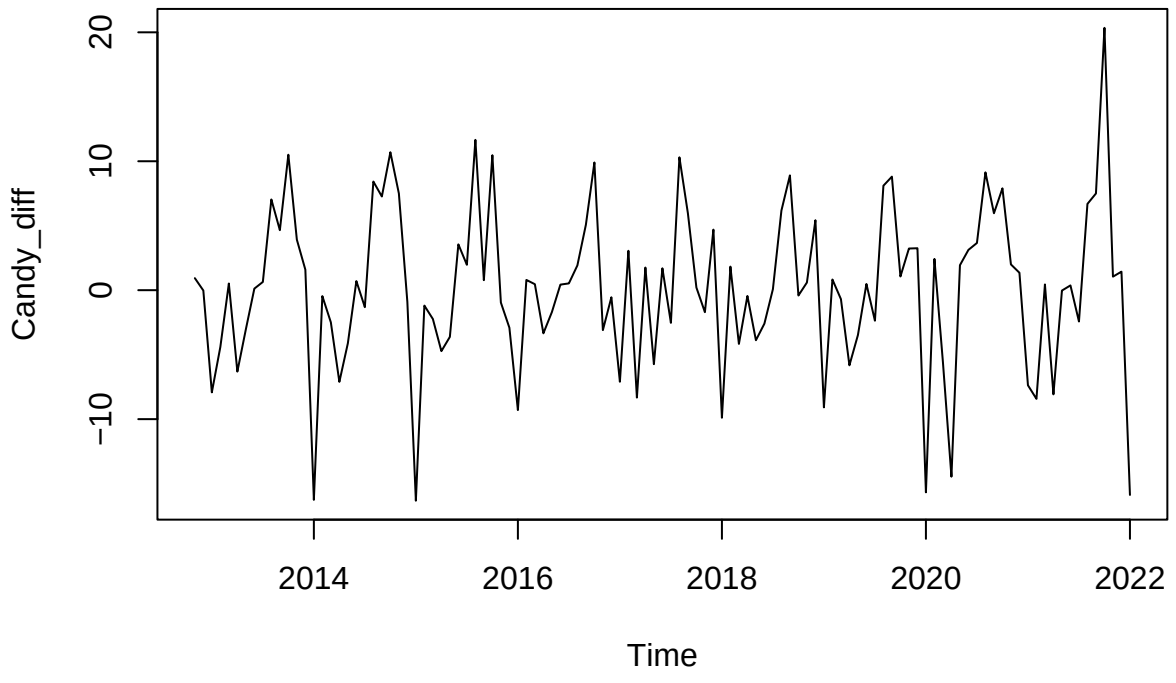
```
#ARIMA

nsdiffs(Window_candy_TS)
```

```
## [1] 1
```

From this we can see that the number of differences in order to make the data stationary is 1.

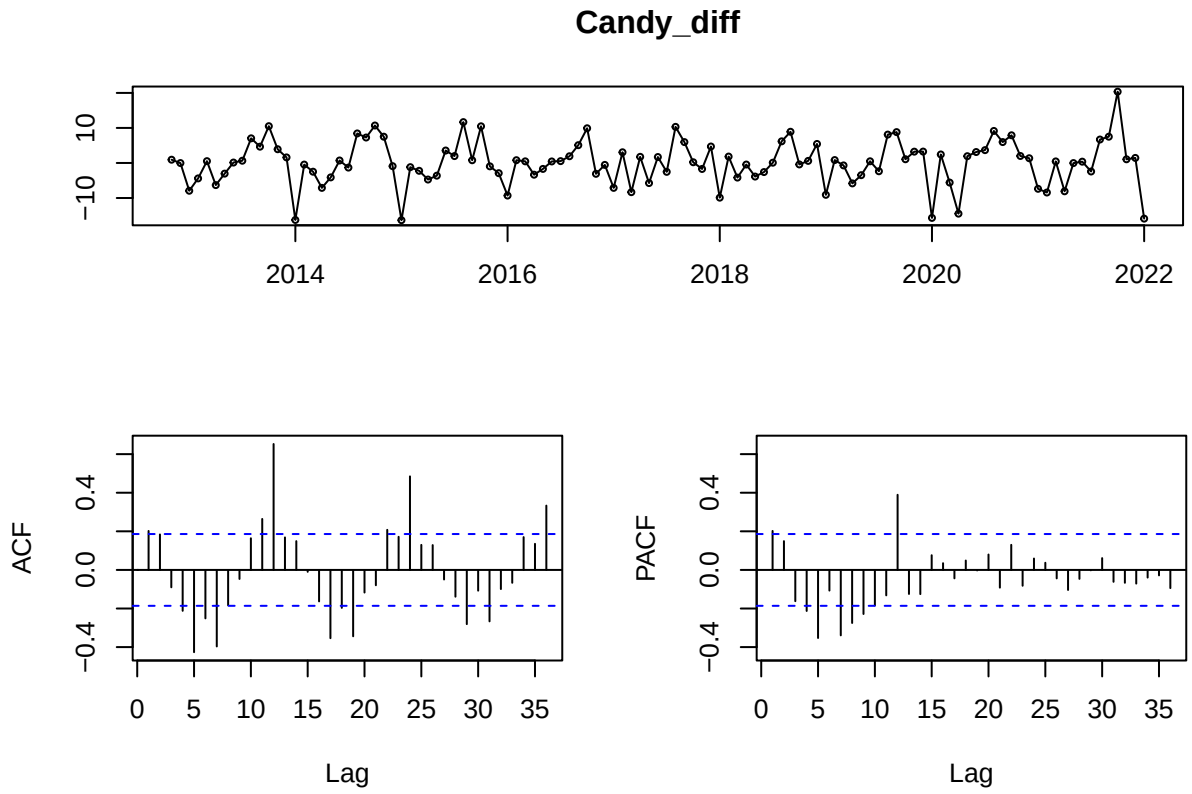
The seasonality component is needed since out time series has a seasonality component.

```
#ARIMA

Candy_diff <- diff(Window_candy_TS, differences=1)
plot(Candy_diff)
```



##This is our plot including the difference of 1.

```
#ARIMA


tsdisplay(Candy_diff)
```




```
#ARIMA


fit1 <- auto.arima(candy_ts)
fit1
```

```
## Series: candy_ts
## ARIMA(1,0,0)(2,1,0)[12] with drift
##
## Coefficients:
##          ar1     sar1     sar2   drift
##       0.6441  -0.3042  -0.2351  0.2083
## s.e.  0.0742   0.1040   0.1099  0.0579
##
## sigma^2 = 14.95:  log likelihood = -301.36
## AIC=612.72   AICc=613.31   BIC=626.18
```

From above, we can see that sigma^2 = 14.95, AIC = 612.72 and BIC = 626.18

```
#ARIMA


fit3 <- auto.arima(Window_candy_TS,trace=TRUE, stepwise = FALSE )
```

```
##
##  ARIMA(0,0,0)(0,1,0)[12]                    : 614.2527
##  ARIMA(0,0,0)(0,1,0)[12] with drift         : 598.8945
##  ARIMA(0,0,0)(0,1,1)[12]                    : 616.2334
##  ARIMA(0,0,0)(0,1,1)[12] with drift         : 596.2502
##  ARIMA(0,0,0)(0,1,2)[12]                    : 616.6361
##  ARIMA(0,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,0)(1,1,0)[12]                    : 616.2156
##  ARIMA(0,0,0)(1,1,0)[12] with drift         : 597.147
##  ARIMA(0,0,0)(1,1,1)[12]                    : 618.2044
##  ARIMA(0,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(1,1,2)[12]                    : 618.617
##  ARIMA(0,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,0)(2,1,0)[12]                    : 617.49
##  ARIMA(0,0,0)(2,1,0)[12] with drift         : 598.7375
##  ARIMA(0,0,0)(2,1,1)[12]                    : 619.6502
##  ARIMA(0,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(2,1,2)[12]                    : Inf
##  ARIMA(0,0,0)(2,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(0,1,0)[12]                    : 578.5893
##  ARIMA(0,0,1)(0,1,0)[12] with drift         : 569.7968
##  ARIMA(0,0,1)(0,1,1)[12]                    : 578.8361
##  ARIMA(0,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(0,1,2)[12]                    : 580.2324
##  ARIMA(0,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(1,1,0)[12]                    : 578.5447
##  ARIMA(0,0,1)(1,1,0)[12] with drift         : 565.7557
##  ARIMA(0,0,1)(1,1,1)[12]                    : 580.3731
##  ARIMA(0,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(1,1,2)[12]                    : 582.0542
##  ARIMA(0,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(2,1,0)[12]                    : 580.1977
##  ARIMA(0,0,1)(2,1,0)[12] with drift         : 567.6842
##  ARIMA(0,0,1)(2,1,1)[12]                    : 582.4073
##  ARIMA(0,0,1)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(2,1,2)[12]                    : Inf
##  ARIMA(0,0,1)(2,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(0,1,0)[12]                    : 569.2631
##  ARIMA(0,0,2)(0,1,0)[12] with drift         : 563.4987
##  ARIMA(0,0,2)(0,1,1)[12]                    : 569.1028
##  ARIMA(0,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,2)(0,1,2)[12]                    : 571.0359
##  ARIMA(0,0,2)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(1,1,0)[12]                    : 569.2359
##  ARIMA(0,0,2)(1,1,0)[12] with drift         : 560.5687
##  ARIMA(0,0,2)(1,1,1)[12]                    : 570.4864
```

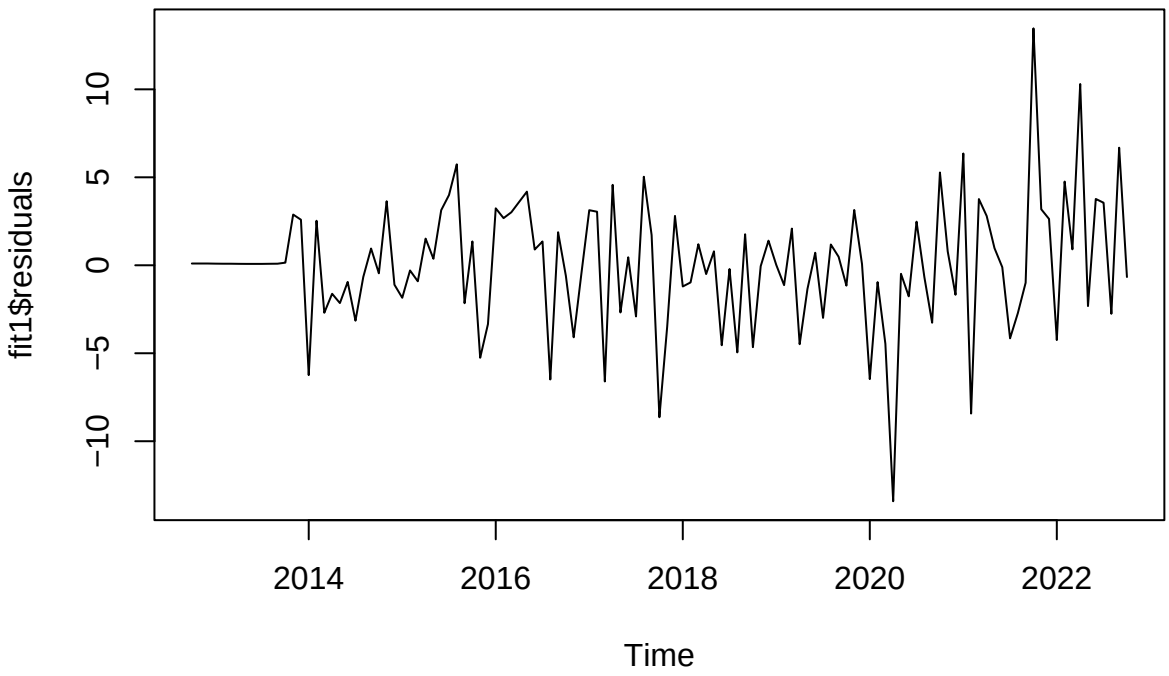
```
##  ARIMA(0,0,2)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,2)(1,1,2)[12]                    : 572.7506
##  ARIMA(0,0,2)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(2,1,0)[12]                    : 571.4252
##  ARIMA(0,0,2)(2,1,0)[12] with drift         : 561.8001
##  ARIMA(0,0,2)(2,1,1)[12]                    : 572.7501
##  ARIMA(0,0,2)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(0,1,0)[12]                    : 564.2611
##  ARIMA(0,0,3)(0,1,0)[12] with drift         : 560.9119
##  ARIMA(0,0,3)(0,1,1)[12]                    : 563.2243
##  ARIMA(0,0,3)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(0,1,2)[12]                    : 565.0771
##  ARIMA(0,0,3)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,3)(1,1,0)[12]                    : 563.4556
##  ARIMA(0,0,3)(1,1,0)[12] with drift         : 557.7127
##  ARIMA(0,0,3)(1,1,1)[12]                    : 564.0822
##  ARIMA(0,0,3)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(2,1,0)[12]                    : 565.6637
##  ARIMA(0,0,3)(2,1,0)[12] with drift         : 559.3016
##  ARIMA(0,0,4)(0,1,0)[12]                    : 564.9765
##  ARIMA(0,0,4)(0,1,0)[12] with drift         : 562.6653
##  ARIMA(0,0,4)(0,1,1)[12]                    : 563.1839
##  ARIMA(0,0,4)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,4)(1,1,0)[12]                    : 563.5343
##  ARIMA(0,0,4)(1,1,0)[12] with drift         : 559.2561
##  ARIMA(0,0,5)(0,1,0)[12]                    : 567.041
##  ARIMA(0,0,5)(0,1,0)[12] with drift         : 564.1867
##  ARIMA(1,0,0)(0,1,0)[12]                    : 560.4046
##  ARIMA(1,0,0)(0,1,0)[12] with drift         : 558.1656
##  ARIMA(1,0,0)(0,1,1)[12]                    : 557.062
##  ARIMA(1,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(0,1,2)[12]                    : 556.7227
##  ARIMA(1,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,0)(1,1,0)[12]                    : 557.8419
##  ARIMA(1,0,0)(1,1,0)[12] with drift         : 554.1541
##  ARIMA(1,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,0)(2,1,0)[12]                    : 559.8259
##  ARIMA(1,0,0)(2,1,0)[12] with drift         : 555.6822
##  ARIMA(1,0,0)(2,1,1)[12]                    : 562.0418
##  ARIMA(1,0,0)(2,1,1)[12] with drift         : 558.1768
##  ARIMA(1,0,0)(2,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(2,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,0)[12]                    : 562.3449
##  ARIMA(1,0,1)(0,1,0)[12] with drift         : 560.3232
##  ARIMA(1,0,1)(0,1,1)[12]                    : 558.9252
##  ARIMA(1,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(1,1,0)[12]                    : 559.7744
##  ARIMA(1,0,1)(1,1,0)[12] with drift         : 556.3705
##  ARIMA(1,0,1)(1,1,1)[12]                    : Inf
```

```
##  ARIMA(1,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,1)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(2,1,0)[12]                    : 561.7711
##  ARIMA(1,0,1)(2,1,0)[12] with drift         : 557.9455
##  ARIMA(1,0,1)(2,1,1)[12]                    : 563.9685
##  ARIMA(1,0,1)(2,1,1)[12] with drift         : 560.4787
##  ARIMA(1,0,2)(0,1,0)[12]                    : 564.0932
##  ARIMA(1,0,2)(0,1,0)[12] with drift         : 561.8723
##  ARIMA(1,0,2)(0,1,1)[12]                    : 561.1159
##  ARIMA(1,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,2)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,2)(1,1,0)[12]                    : 561.9663
##  ARIMA(1,0,2)(1,1,0)[12] with drift         : 558.4588
##  ARIMA(1,0,2)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,2)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(2,1,0)[12]                    : 563.9871
##  ARIMA(1,0,2)(2,1,0)[12] with drift         : 559.9179
##  ARIMA(1,0,3)(0,1,0)[12]                    : 565.2932
##  ARIMA(1,0,3)(0,1,0)[12] with drift         : 562.8614
##  ARIMA(1,0,3)(0,1,1)[12]                    : 563.0312
##  ARIMA(1,0,3)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,3)(1,1,0)[12]                    : 563.536
##  ARIMA(1,0,3)(1,1,0)[12] with drift         : 559.4505
##  ARIMA(1,0,4)(0,1,0)[12]                    : 567.142
##  ARIMA(1,0,4)(0,1,0)[12] with drift         : 564.8129
##  ARIMA(2,0,0)(0,1,0)[12]                    : 562.3127
##  ARIMA(2,0,0)(0,1,0)[12] with drift         : 560.3192
##  ARIMA(2,0,0)(0,1,1)[12]                    : 558.9159
##  ARIMA(2,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(2,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(2,0,0)(1,1,0)[12]                    : 559.7676
##  ARIMA(2,0,0)(1,1,0)[12] with drift         : 556.3704
##  ARIMA(2,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(2,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(2,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(2,0,0)(2,1,0)[12]                    : 561.759
##  ARIMA(2,0,0)(2,1,0)[12] with drift         : Inf
##  ARIMA(2,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(0,1,0)[12]                    : 564.4078
##  ARIMA(2,0,1)(0,1,0)[12] with drift         : 562.4507
##  ARIMA(2,0,1)(0,1,1)[12]                    : 561.1317
##  ARIMA(2,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(2,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(2,0,1)(1,1,0)[12]                    : 561.9841
##  ARIMA(2,0,1)(1,1,0)[12] with drift         : 558.6319
##  ARIMA(2,0,1)(1,1,1)[12]                    : Inf
##  ARIMA(2,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(2,1,0)[12]                    : Inf
```

```
##  ARIMA(2,0,1)(2,1,0)[12] with drift         : 560.1475
##  ARIMA(2,0,2)(0,1,0)[12]                    : 566.089
##  ARIMA(2,0,2)(0,1,0)[12] with drift         : 563.1382
##  ARIMA(2,0,2)(0,1,1)[12]                    : 563.3754
##  ARIMA(2,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,2)(1,1,0)[12]                    : 564.2091
##  ARIMA(2,0,2)(1,1,0)[12] with drift         : 560.1002
##  ARIMA(2,0,3)(0,1,0)[12]                    : 567.3445
##  ARIMA(2,0,3)(0,1,0)[12] with drift         : 564.098
##  ARIMA(3,0,0)(0,1,0)[12]                    : 564.219
##  ARIMA(3,0,0)(0,1,0)[12] with drift         : 561.827
##  ARIMA(3,0,0)(0,1,1)[12]                    : 561.1288
##  ARIMA(3,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(3,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(3,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(3,0,0)(1,1,0)[12]                    : 561.9824
##  ARIMA(3,0,0)(1,1,0)[12] with drift         : 558.4254
##  ARIMA(3,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(3,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(3,0,0)(2,1,0)[12]                    : 564.012
##  ARIMA(3,0,0)(2,1,0)[12] with drift         : 559.8617
##  ARIMA(3,0,1)(0,1,0)[12]                    : Inf
##  ARIMA(3,0,1)(0,1,0)[12] with drift         : Inf
##  ARIMA(3,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(3,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(3,0,1)(1,1,0)[12]                    : Inf
##  ARIMA(3,0,1)(1,1,0)[12] with drift         : 560.4603
##  ARIMA(3,0,2)(0,1,0)[12]                    : Inf
##  ARIMA(3,0,2)(0,1,0)[12] with drift         : Inf
##  ARIMA(4,0,0)(0,1,0)[12]                    : 566.2388
##  ARIMA(4,0,0)(0,1,0)[12] with drift         : 563.3861
##  ARIMA(4,0,0)(0,1,1)[12]                    : 563.3083
##  ARIMA(4,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(4,0,0)(1,1,0)[12]                    : 564.0207
##  ARIMA(4,0,0)(1,1,0)[12] with drift         : 559.6753
##  ARIMA(4,0,1)(0,1,0)[12]                    : Inf
##  ARIMA(4,0,1)(0,1,0)[12] with drift         : 565.6208
##  ARIMA(5,0,0)(0,1,0)[12]                    : 567.6812
##  ARIMA(5,0,0)(0,1,0)[12] with drift         : 565.4329
##
##
##
##  Best model: ARIMA(1,0,0)(1,1,0)[12] with drift
```

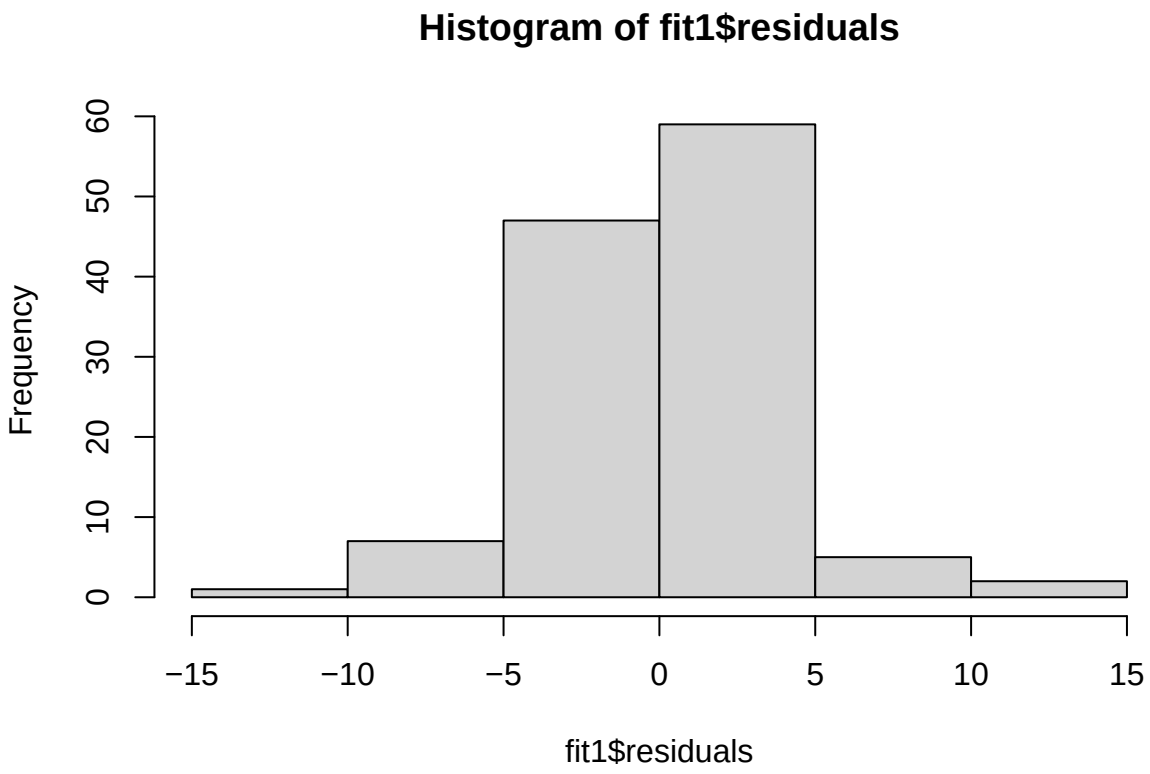
Based on the these auto arima functions run, we can see that the best model is ARIMA(1,0,0)(2,1,0)[12] with drift

```
#ARIMA


plot(fit1$residuals)
```



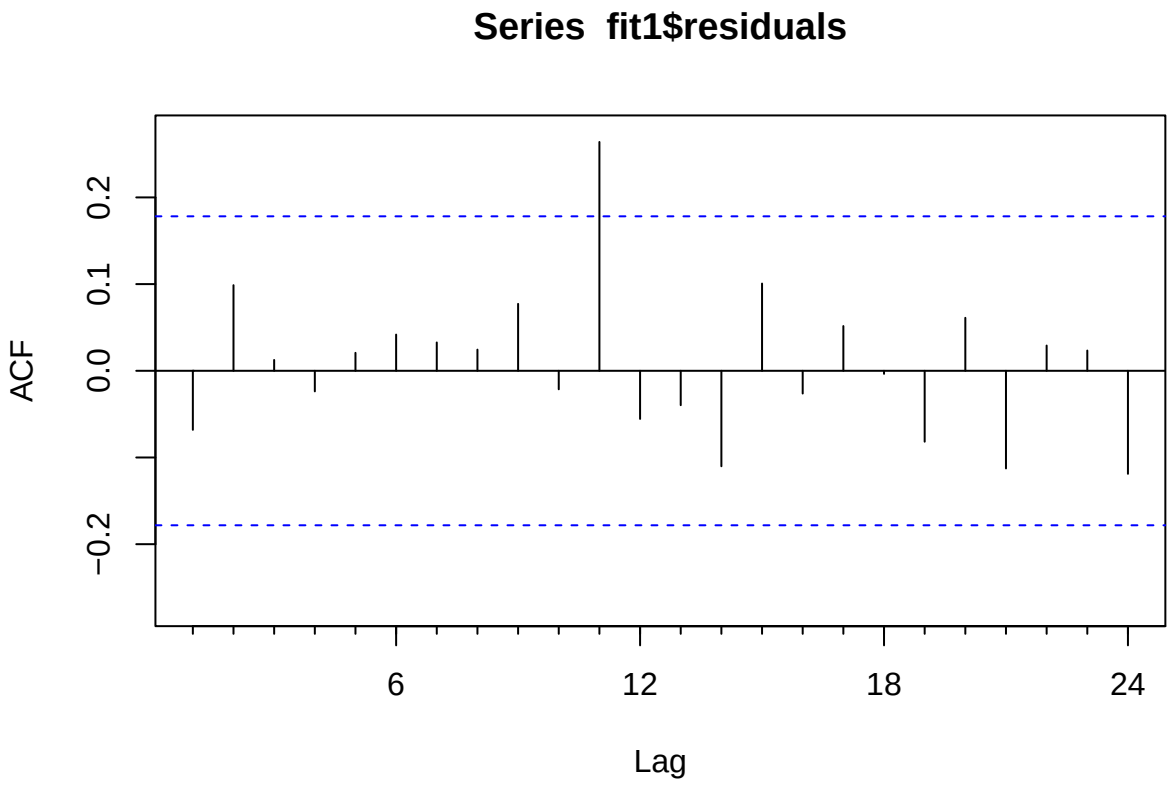
These residuals looks very good since it appears to be white noise - There appears to be no seasonality or trend.

```
#ARIMA


hist(fit1$residuals)
```



This histogram plot seems to indicate that the data is normally distributed which is good for our residuals.

```
#ARIMA


Acf(fit1$residuals)
```

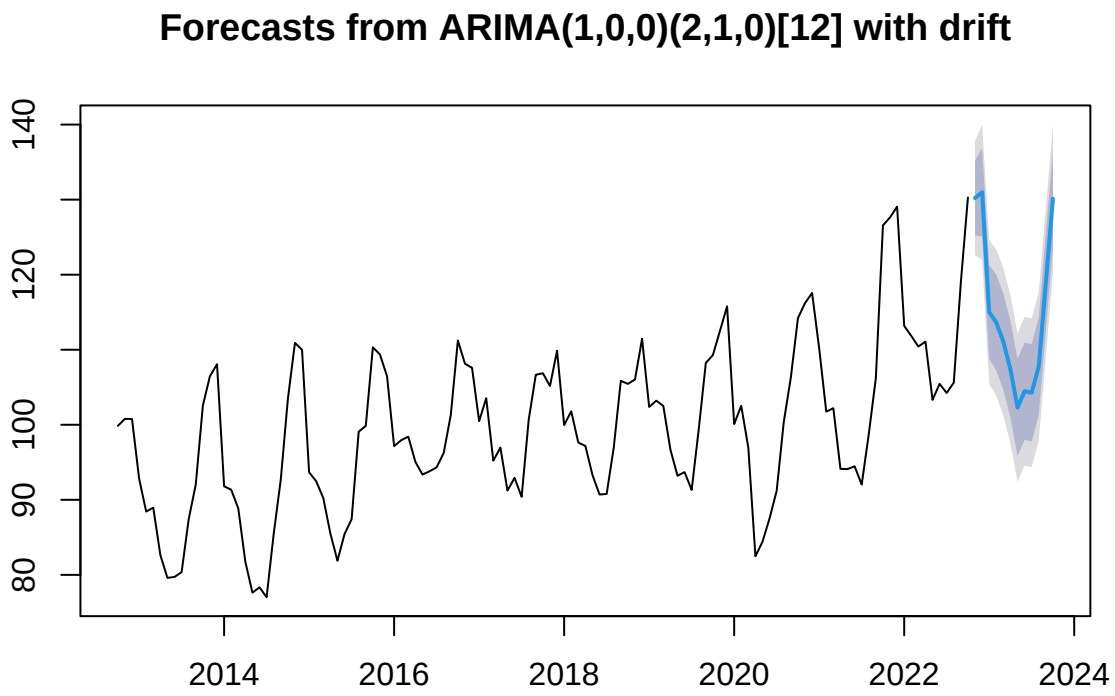


This plot indicates that this plot is indeed the best because there is no trend or seasonality in the data. There values are also not statistically significant except for one and that appears to be a mistake.

```
#ARIMA

accuracy(fit1)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.004444951 3.602064 2.583457 -0.1100037 2.571757 0.5881103
##                      ACF1
## Training set -0.06809078
```

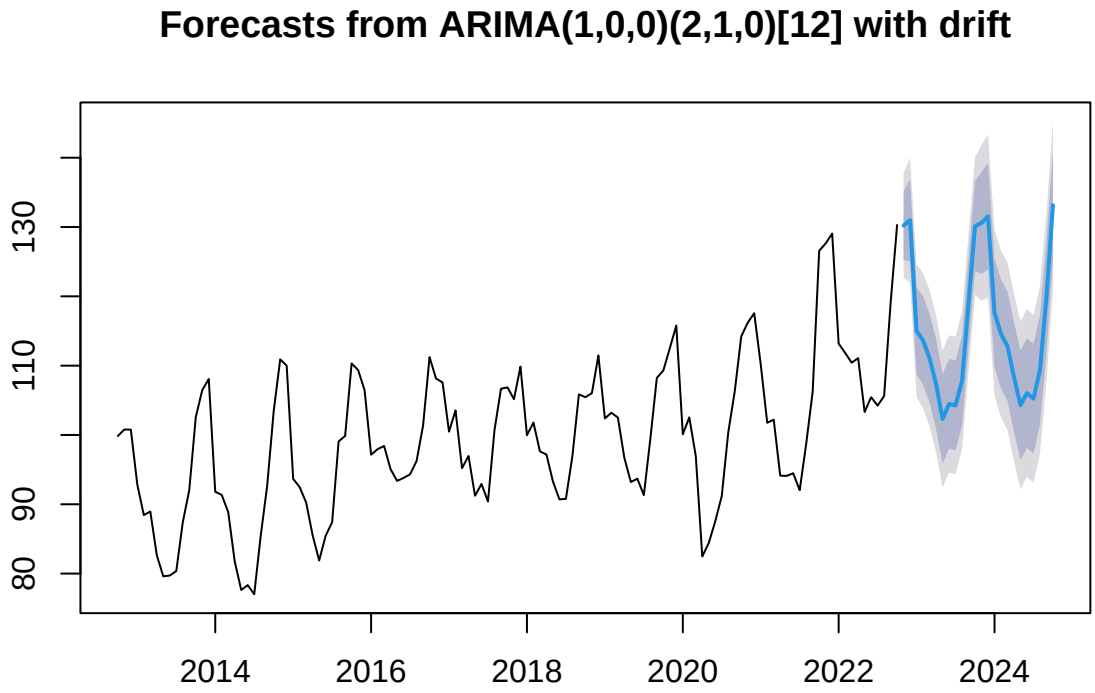
```
#ARIMA


oneyear_arima <- (forecast(fit1, h=12))
plot(oneyear_arima)
```




```
#ARIMA


oneyear_arima
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Nov 2022       130.2152 125.25972 135.1707 122.63645 137.7939
## Dec 2022       130.9781 125.08373 136.8725 121.96344 139.9928
## Jan 2023       115.0259 108.78339 121.2685 105.47879 124.5731
## Feb 2023       113.6240 107.24263 120.0054 103.86452 123.3835
## Mar 2023       111.0655 104.62735 117.5036 101.21922 120.9117
## Apr 2023       107.3725 100.91103 113.8341  97.49052 117.2546
## May 2023       102.3049  95.83376 108.7761  92.40812 112.2018
## Jun 2023       104.4700  97.99476 110.9451  94.56700 114.3729
## Jul 2023       104.2670  97.79014 110.7439  94.36150 114.1725
## Aug 2023       107.8069 101.32940 114.2845  97.90040 117.7135
## Sep 2023       119.0309 112.55302 125.5087 109.12386 128.9378
## Oct 2023       130.1239 123.64593 136.6018 120.21671 140.0310
```

```
#ARIMA


twoyear_arima <- (forecast(fit1, h=24))
plot(twoyear_arima)
```




```
#ARIMA


twoyear_arima
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Nov 2022       130.2152 125.25972 135.1707 122.63645 137.7939
## Dec 2022       130.9781 125.08373 136.8725 121.96344 139.9928
## Jan 2023       115.0259 108.78339 121.2685 105.47879 124.5731
## Feb 2023       113.6240 107.24263 120.0054 103.86452 123.3835
## Mar 2023       111.0655 104.62735 117.5036 101.21922 120.9117
## Apr 2023       107.3725 100.91103 113.8341  97.49052 117.2546
## May 2023       102.3049  95.83376 108.7761  92.40812 112.2018
## Jun 2023       104.4700  97.99476 110.9451  94.56700 114.3729
## Jul 2023       104.2670  97.79014 110.7439  94.36150 114.1725
## Aug 2023       107.8069 101.32940 114.2845  97.90040 117.7135
## Sep 2023       119.0309 112.55302 125.5087 109.12386 128.9378
## Oct 2023       130.1239 123.64593 136.6018 120.21671 140.0310
## Nov 2023       130.6058 123.25537 137.9562 119.36428 141.8473
## Dec 2023       131.5482 123.86490 139.2316 119.79758 143.2989
## Jan 2024       117.6141 109.79684 125.4314 105.65861 129.5696
## Feb 2024       114.5620 106.68977 122.4341 102.52249 126.6014
## Mar 2024       112.7877 104.89290 120.6826 100.71362 124.8619
## Apr 2024       108.3628 100.45859 116.2670  96.27435 120.4513
## May 2024       104.2936  96.38545 112.2017  92.19915 116.3880
## Jun 2024       106.0342  98.12451 113.9440  93.93735 118.1311
```

```
## Jul 2024       105.2377  97.32733 113.1481  93.13982 117.3356
## Aug 2024       109.3693 101.45862 117.2800  97.27097 121.4676
## Sep 2024       119.8675 111.95676 127.7783 107.76904 131.9660
## Oct 2024       133.1484 125.23761 141.0593 121.04987 145.2470
```

We will see that in October 2023(a year from this point in time) the prediction is giving us 130 and in two years time(October 2024) we are getting 133.

Overall we can see that this forecatsing technique is very good; the residuals are random and not significant and we can see the forecast is incorporating the positive trend as well as seasonality.